From: Ed Lowry
To: USDoJ Antitrust Div
Date: 1/24/02 9:15am
Subject: Micrsoft Settlement

[Text body exceeds maximum size of message body (8192 bytes). It has been converted to attachment.]

From: Edward S. Lowry
7 Alder Way
Bedford Mass 01730
781 276-4098
eslowry@alum.mit.edu
http://www.ultranet.com/~eslowry

January 24, 2002

To: Hon. Colleen Kollar-Kotely
United States District Court for the District of Columbia
333 Constitution Avenue NW
Washington DC 20001

cc: U.S. Department of Justice, Antitrust Division

Subject: Microsoft settlement, blunt their complexity weapon

Dear Justice Kollar-Kotely:

Over 30 years of anticompetitive behavior in the software industry have been far more destructive than the proceedings in the Microsoft case have suggested. Microsoft's abuse of its competitors is a minor part of larger destructive effects.

The designers of antitrust legislation probably never contemplated an industry where market competition could be so easily thwarted. The large inherent complexity of software makes changing vendors difficult. This creates a business incentive for all vendors to make software more complex than it needs to be. Competition is further weakened by incentives for vendors and others to claim they are trying to simplify while actually pursuing contrary policies, thus discouraging real efforts at large scale simplification. It is further weakened by the existence of so many people with prestige or profit incentives to complicate other peoples' lives. I have decades of experience working on simplification and observing deliberate obstruction of large scale simplification by prestigious organizations. MY EXPERIENCE HAS BEEN CONSISTENT WITH THE VIEW THAT HIDING THE TRUTH ABOUT THE POTENTIAL FOR SIMPLIFICATION HAS BEEN A HIGH PRIORITY AMONG SOFTWARE LEADERS FOR DECADES.

This pattern makes correcting for past and prospective anticompetitive behavior in software both difficult and necessary. Your authority to protect the public interest is now precious to society at large. I recommend that you take no action which limits the exercise of your authority until remedies which you set in motion have been demonstrated to be effective. I also recommend that your attention focus largely on reducing the effectiveness of needless complexity as a tool to thwart competition.

The consumer need for simplicity in computing has always been huge and obvious. Millions of people every day bear enormous burdens of needless complexity which are a direct result of disregard for the public interest by the software industry.

THERE ARE ONLY A FEW DOZEN PEOPLE ANYWHERE WHO HAVE USED PROGRAMMING LANGUAGE SEMANTICS FOR NON-TRIVIAL APPLICATIONS AS ADVANCED IN ALLOWING FOR SIMPLICITY OF EXPRESSION AS WHAT IBM DESIGNED OVER 25 YEARS AGO. The gross failure of marketplace competition to provide incentives to make reasonable simplifications is a matter of demonstrable fact.

In part to provide some confirmation of the 25 year delay, I asked U.S. Senator John Kerry to make an inquiry related to these issues. The main question is whether anyone working in the

Federal Government has experience using software technology which allows for simplicity as advanced as what was designed over 25 years ago. He initiated that inquiry in December 2000 and hasn't reported finding anyone yet. The lack of such experience among technical advisors can be demonstrated fairly easily using the attached programming examples. Failure to take first steps in simplifying software has blocked a series of potential simplifications.

Results of decades of deliberate anticompetitive resistance to large scale simplification in software include:

- deficient math and science education due to failure to express precise information precisely.
- a flood of needless and burdensome complexity going into schools as educational technology.
- users entangled in proprietary complexity for decades.
- massive degradation of the quality of technical information: accessibility, usability, clarity, interoperability, ductility.
- a probable contributing factor to crash of KA801 in Guam, August 1997, 226 dead.
- a possible contributing factor to 20% of US casualties in the Gulf War.
- failure of the FAA to upgrade its air traffic safety systems.
- massive government waste such as the IRS modernization effort.
- mismatch between employment skills needed and those available.
- major "innovations" are "square wheel" unreasonable.
- political leaders dependent on technical advisors who are over 25 years behind the leading edge on simplicity issues.
- burdensome technological instability.
- illusions of progress and illusions of effort to make progress.
- large learning loads for skills of ephemeral value. Several hundred feet of bookshelf space (e.g. at Barnes and Noble) to tell people how to tell their computers what they want.
- massive vulnerability to computer viruses.

For decades there has been almost nothing standing between the public and this kind of abuse. At present there is almost nothing to prevent still more decades of the same. Remedies appropriate to the scale of past damage and the prospect of future damage are needed. That requires assessing the scale of the damage and I doubt there has been any realistic effort to do so. A great deal depends on your response. It doubt that Microsoft will agree to anything potent enough to correct for this overall pattern.

The most serious damage of anti-competitive behavior in software has probably been to obstruct improvement in technical education. On a massive scale, educators teach how to arrange pieces of information. However, by failing to understand simplification, they have almost no idea what is a reasonable structure for pieces of information. One result is a failure to express precise information in precise ways, and that imposes a large needless burden on students. Imagine cities where the architects and builders had never seen a reasonably shaped brick. Both technical education and software technology are mired in a comparable state of reasonableness today.

For over 30 years I have been working to clarify how to make software and other technical description simple, mainly at IBM and Digital Equipment Corporation. I have observed a high level of confidence among software leaders that there would be no accountability for opposing large scale simplification. I can elaborate on all of the above from long experience.

The anticompetitive behavior has distorted antitrust proceedings

by keeping participants in a state of technical ignorance. By accepting prevailing stunted understanding of simplicity in software from people who are demonstrably 25 years behind the leading edge, the Department of Justice has probably tended to see existing ongoing dependency relationships between vendors and users of software technology as largely the result of complexity which is inherent in the technology. That is unrealistic because so much of the complexity has been caused by deliberate actions by vendors. It would also appear that the potential for competitiveness is more limited than it actually is. Such views would have probably biased Department of Justice toward non-intervention and obscured the potential for judicial remedies.

The proposed settlement agreement itself looks like a hollow gesture resulting partly from poor technical understanding at DoJ. It focuses on assuring equitable access to the market for "middleware". Middleware as defined amounts to software for executing special purpose languages. The need for such software would be greatly altered and diminished if serious efforts to simplify were made. Accepting such an agreement would put the court in a position of depending on continued large scale anticompetitive behavior in the industry in order to place minor restraints on Microsoft.

The best available analysis of the fine structure of information supports the view that almost all computerized information is represented in ways that are unreasonable in much the same way that square wheels are unreasonable. No other field of technology has been afflicted by such perversity. No other community of technologists have been so oblivious about their most basic structures. The consequences have disastrously damaged information quality and disabled human minds. It has been largely unnoticed because people have no other experience. While that analysis [see "Toward Perfect Information Microstructures" on my web site] may be viewed as unverified, not even the beginnings of a technically sound alternative analysis exists. It includes easily understood reasons why simplification allows "middleware" or special purpose languages to be displaced by much more versatile general purpose language. Patents now owned by Compaq Corporation have helped to prevent corrective measures.

I hope that you will assess the effectiveness and destructiveness of the anticompetitive behavior. You could ask the parties to the case: who among them has working familiarity with software for simplicity as advanced as IBM developed over 25 years ago (using the test below). Noting the above paper arguing that the basic structures of software technology are now "square wheel" unreasonable, you could ask whether any can identify a sound presentation of an alternative technical view. In addition the intent should be assessed. To what extent has obliviousness to needless complexity been used as a concealed weapon against competitors? My story supports such intent.

I hope that you will be bold in developing remedies and prepared for some contention. Major social upheavals have sometimes accompanied simplifications in law and religion. Publicly asking embarrassing questions like the above could help a lot. Such questions have been passionately evaded for many years. Even a U.S. Senator has had difficulty eliciting answers.

Microsoft could be required to support large scale simplification in a variety of ways. They could be required to respond to or financially support challenges to understand simplification like the $25K offer on my web site. Another approach would be to require that they incorporate a conceptually self-contained interface into Windows which provides broad capability with simplicity close to the best available understanding of how to do

so. It could be aimed specifically at educational use, at least initially. Rolling back the unwelcome intrusion into education of mysteriousness and technological instability could be major achievable goals of such an effort. It could be partly defined in terms of avoiding damage to quality of user information.

I can provide additional supporting documents and other assistance. Let me know if I can help .

Yours respectfully

Ed Lowry

****

SHANNON Examples Compared with SEQUEL 2

To assist in validating a 25 year delay in improving simplicity of expression, and raising quality standards in future efforts, the following list of example expressions are provided. They indicate what degree of simplicity and clarity is achievable in a multi-purpose language and roughly what was known to be achievable in 1974 as recorded in: "PROSE Specification" by E. S. Lowry, IBM Poughkeepsie Laboratory Technical Report TR 00.2902. It was dated Nov 1977 but it was distributed within IBM in Dec 1974.

These examples are translated from the first 10 examples given for Sequel 2 (now SQL) in the IBM Journal of R&D, Nov 1976. For the first 10 expressions Sequel 2 (a specialized data base language) uses 130 tokens. Shannon (a multi-purpose language) uses 99 tokens. The original Sequel 2 code is omitted as irrelevant. The significant comparisons would be with C++, Java, Ada, Cobol, etc.

Expression 1.

English: Names of employees in Dept. 50

Shannon: name of employee of dept(50)

Expression 2.

Eng: All the different department numbers in the Employee table.

Shan: dept_no of employee condense

Expression 3.

Eng: Names of employees in Depts. 25, 47 and 53.

Shan: name of employee of every dept where 25 or 47 or 53

Expression 4.

Eng: Names of employees who work for departments in Evanston.

Shan: name of employee of dept of Evanston

Expression 5.

Eng: List the employee number, name and salary of
employees in Dept. 50, in order of employee number.

Shan: for employee of dept(50) minfirst empno
show(empno, name, salary)

Expression 6.

Eng: Average salary of clerks.

Shan: average (salary of clerk)

Expression 7.

Eng: Number of different jobs held by employees in Dept.50

Shan: count job of employee of dept(50) condense

Expression 8.

Eng: List all the departments and the average salary of each.

Shan: for dept show(it, average(salary of its employee))

Expression 9.

Eng: Those departments in which the average employee salary is
less than 10,000.

Shan: dept where average(salary of its employee) < 10000

Expression 10.

Eng: The departments that employ more than ten clerks.

Shan: dept where count(its clerk) > 10